Generative Sound Models

Lonce Wyse
Institute for Infocomm Research
Singapore, 119613
lonce@i2r.a-star.edu.sg

Abstract

An overview of generative sound models is presented. We discuss the benefits they offer in a variety of media contexts including indexing and retrieval, compression, sonification, traditional media and interactive media production. We examine ways in which working with sound models differs from working with sound. Model design is identified as an outstanding research issue, and several strategies for addressing this challenge are presented.

Keywords

Sound modeling, audio semantics, audio indexing and retrieval, audio postproduction, sonification, interactive sound design.

1. Introduction

Sound is usually used in media in an unstructured form, as a string of numbers representing discrete samples in time. Even when compressed, high-quality audio can take up large amounts of memory and bandwidth. The unstructured form also provides no information about how the sound was generated - information that would be useful for indexing and retrieval as well as for interactively controlling the sound in ways that are germane to its source. Sound models address these issues.

A generative sound model is an algorithm for synthesizing a class of sounds under parameterized control, typically in real time. Media modeling is attracting interest because of the growth in realtime interactive applications such as computer games where the potential media experience is far too open-ended for all possibilities to be prerendered. For interactive media, objects are modeled so that they can respond to a variety of input from their environment and render themselves appropriately. For example, a single 3D "Shrek" character model can run, jump, bend over, and generate facial expressions that are within its range of potential behavior - enabled, but not specifically envisioned by the model designer.

There are many applications areas than can benefit from sound models. Most of the benefit to applications stem from three key advantages that models offer over traditional "flat" representations of sound:

a) they use much less memory (or bandwidth),
b) they afford realtime flexibility and interactivity,
c) they can function as description of the sounds they generate.

2. Applications

Sound models are an obvious solution for media that is itself interactive such as computer games. However, even when the media product is fixed as in television and film, the production process can still reap great benefits from the flexibility inherent in models. For example, sound for films is almost never recorded at the same time and location as the scene is shot, but is instead added later in a recording studio. In an elaborate process called Foley, sound effects are created by Foley artists who watch a film once for all the incidental sounds that are needed, and then armed with an assortment of noise making materials, surfaces for walking on, etc., recreate the sounds for the scene as they watch the film to for proper sound and synchronization.

The Foley process is highly interactive and realtime, but is expensive in human and studio resources. With an adequate set of sound models and appropriate physical controllers, Foley could be done in a desktop environment (Cook 2002), thereby adding tremendous value to the audio production process, rather than to the delivered media itself as it does for computer games.

Graphics modeling has been the focus of intense research and development over the past decade, and has been a key enabling factor in the establishment of the massive computer game industry, driving computer hardware innovation, extending the visual capabilities of film with 'special effects' and providing for a new genre of computer animated films. Audio modeling has received far less attention. Tools do not exist yet that put sophisticated modeling capabilities in to the hands of sound designers that are equal to the standard tool set of every graphics designer's workbench.

The history of twentieth century music is also a story of an ever greater acceptance and use of all sound material in both classical and popular genres. Synthesizers and computers are standard tools in music composition and production, but the full and equal access to arbitryary sound control is still limited (Wyse, 2003).

Wyse, L. (2005). Generative Sound Models. *Proceedings of the Multimedia Modeling 2005 Conference.* Melbourne, Australia. 370-377.

Until the issues in sound models design are resolved, "interactive" audio will remain a technique of clever patterns of triggering prerecorded audio, and will not bestow the bandwidth and flexibility benefits to the applications that they have the potential to offer.

3. Models and Synthesizers

The main properties of a sound model are

a) a specific range of sound the model is capable of producing,
b) a set of parameters for control,
c) the paths through the range of sounds that determine what we shall call the *behavior* of the model, that are traversed as the parameters change.

Models consist of components that take on different significance depending upon the application. Broadly, they include a synthesizer, a set of parameters for control, and structure for mapping between control parameters and synthesis elements. The output is sound that that can be perceived by humans or analyzed by computers to produce features that are used to identify, classify, label and compare sounds (Figure 1).

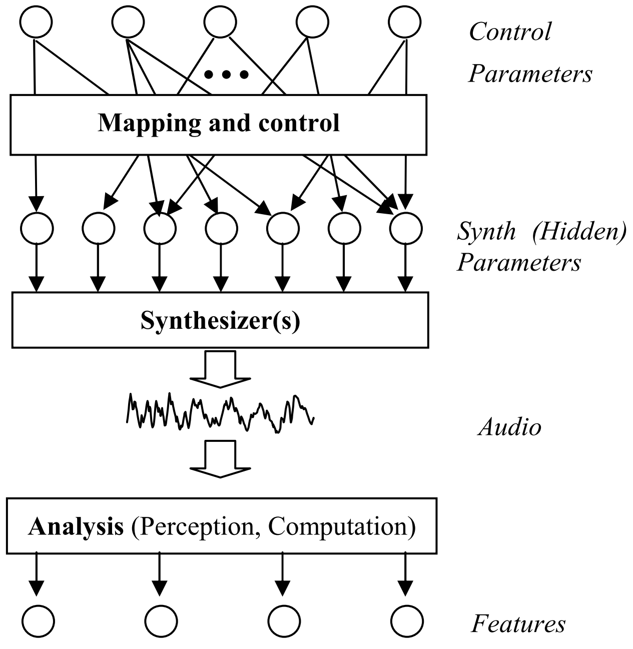

Figure 1. Conceptual components of a Sound Model.

Models can span a wide range of complexity, power and identity. There is no clear dividing line between a sound model and a synthesizer, for example, which is itself a parameterized sound generating machine. Synthesizers are built from elements such as oscillators and filters, and the (often massive number of) parameters are the same for each and every sound the synthesizer produces. The signal processing and synthesis levels for sound are what color, lines and polygons are to graphics. A model-level description for sound is what characters and objects are to graphics.

The design goal of a typical synthesizer construction is an architecture that enables the widest possible range of sounds. The design goal for sound modeling is to construct a narrowly constrained range of sounds with a distinctive behaviors and identity. A synthesizer exposes a single (usally large) set of parameters that are used to control all of its sounds, and generally have a straight forward effect on a specific signal processing component of the synthesizer. A sound model has sound-specific parameters that, due to the mapping and control structures, might have a complex coordinated effect across many of the signal processing components of the synthesizer, or an effect on the control architecture itself.

Another important concept in the distinction between a synthesizer and a sound model is that of an "event". In line with traditional musical concepts, we make a distinction between the source of control signals and events (an instrument player) from that of the sound synthesizer (a musical instrument). A sound model often includes event pattern generators, frequently initiating events on a time scale impossible for human gestures to achieve. Of course, there remains an external user of sound models that also generates control signals including events.

The parameter mapping, control architecture, and event patterning implemented "under the hood" by potentially complicated and multiple synthesis algorithms, relationships between low level synthesis parameters, dynamics, and nonlinearities, give the model its identifying characteristics. A car engine, for example, involves multiple sound sources related by different kinds of mechanical and acoustic couplings producing a wide variety of noises under fairly simple realtime parametric control (a gas pedal and engine "work load"). Despite the internal complexity, we can hear underlying model "invariants", for instance that an engine does not change size, even though a wide range of sounds are produced over the control parameter range. Invariant structures defining behavior are unique to each model, not generic and customizable by a single set of parameters, and they define morphological relationships between different sounds. They are also what will allow models to be associated with semantics that will discussed later in the paper.

4. Models and Sounds

For any given sound, there are an unlimited number of algorithms, and thus sound models, that could generate it.

As an example of the many-to-one issue in sound modeling, consider the frequency modulation (FM) technique of sound synthesis (Equation 1) which in a simple case, uses one sinusoidal waveform oscillator (the "modulator") to modulate the frequency of a second (the "carrier"). The technique results in a series of sine waves at different amplitude levels centered at the carrier and spaced out in frequency by an interval corresponding to modulating frequency.

$$fm(t) = A\sin(2\pi f_c t + [I\sin(2\pi f_m t)]) \quad (1)$$

Because the resulting sound is just a sum of sine waves at different frequencies, the sound could just as well be modeled by a bank of oscillators, one per sine wave in the signal.

If generating a single sound were the goal, it wouldn't matter which signal model was used. It is because we want to design behavior under parametric control that the choice matters and is the reason why there can be no such thing as a super-synthesizer general purpose box that can synthesize any sound given the right parameter settings.

As mentioned above, a sound model is defined not just by the sounds in its range, but also by how the sounds change under parametric manipulation. The same range of sounds can be covered by two entirely different models because of how the range of sounds is traversed depending upon the parameter changes.

Consider again the simple FM and additive algorithms. For any sound the FM algorithm produces under a fixed set of parameters, there exists a parameter setting for the additive model that can generate the same sound. However, each of these two algorithms affords a different number of parameters which control very different kinds of behavior – different trajectories through their range of sounds. An oscillator has two parameters; a frequency and an amplitude (ignoring phase for simplicity). The FM algorithm is built of two oscillators and so is controlled with 4 parameters, while the additive model has twice as many parameters as oscillators. As the frequency of the modulator in the FM algorithm is changed, it affects all of the (perhaps dozens of significant) resulting sinusoidal components in the signal. Thus, in order for the additive model to generate the same signal, many (perhaps dozens) of parameters must be manipulated in a coordinated fashion.

Because the additive synthesis model covers a wider range of sounds than the FM model, then given two sounds, a "source" and a "target", each in the range of both models, the additive model has many more smooth paths it can take between the source and the target.

5. Modeling as compression

Modeling can also be seen and used as an extreme form of audio compression (Scheirer, 2001). A stream of parameters for a model is an encoded version of the generated audio data, and the model functions as a decoder.

Standard compression schemes such as those defined by the standards organization MPEG (Moving Pictures Expert Group), are universal in the sense that there is one encoder/decoder pair that is used for any signal. Generative models on the other hand are designed for narrow classes of sounds. Standard lossy schemes may work somewhat better on some classes of signals than on others, but they are designed to minimize such bias.

Because generative models are not universal representations, the encoding stage requires a model identification step in addition to the signal-to-code step. Furthermore, the constraints on the range and behavior of a typical model are such that the technique will almost certainly be lossy, and the severity of the loss will be highly signal-dependent. Finally, parameterizing models to match signals is an open problem, one that will be discussed in more detail later in the paper.

Despite these drawbacks, models function with extreme efficacy as data reduction schemes under certain conditions:

1) When the class of sounds to be coded is known and constrained (for example, musical "beats" for popular songs (Wyse, Wang, Zhu 2003)).

2) When models can be used to create the sound in the first place (for example in animation and computer games).

6. Semantics and Sound Models

Locating and labeling information in media stores has become an important aspect of media management because of the volumes of data that are the result of the digitization of media production for both professionals and consumers. In addition to their more obvious uses in media production, generative sound models have properties that make them potentially useful in this semantic analysis task.

To address human-centirc means of identifying and searching for material, the objective is a semantic labeling. It is straightforward to extract certain signal level features from an audio stream such as spectral basis vectors (Casey 2001), spectral centroid, zero crossings, pitch, or measurements of noise. Typically, semantic audio analysis is based on a learned association between these low-level signal features and labels provided by a "supervisor", or based on an association with data from another media stream such as video.

Of course, semantics aren't actually "in" the data, but with prior knowledge of a usage context, and by making certain assumptions (e.g. skies are grey or blue and occupy the upper part of a frame) some headway has been made in associating signal features with semantics especially in the realm of graphics. The semantics of non-speech sound, however, tend to be even more flexible and have a more tenuous relationship to the world of objects and events than do graphics. Humans depend more on what an object looks like than how it sounds to identify and define an object. Many quite different objects or events can generate the same sound, but if two objects look the same they are likely to be considered to be the same. A single object is also usually capable of generating many different kinds of sounds.

6.1 Generative models for semantic labeling

Since a model represents a class of sounds – the range that it is capable of generating – so a model can be used for classification. If a test sound is close in feature space to a sound that falls within the range of a particular model, then we can label the test sound with the model.

The technique is related to speech processing with Hidden Markov Models (HMM). An HMM represents a sequence of sounds with a set of states, transition probabilities between states, and a distribution of outputs for each state, where outputs represent sound features. A single HMM represents a class of sounds as a sequence of features that can be generated given a set of states and their particular statistical parameterization. Each sound in the class has an associated likelihood of being generated by the model. With a collection of models parameterized for different classes (different spoken words, for example), a target sound is presented for identification and a likelihood score is computed for each model. The model that produces the highest likelihood score for the sound is chosen as the identifying class.

These models are often termed "generative" because they are capable of statistically generating a sequence of features that correspond to a sound. They are not directly used to synthesize sound because the features that represent the sound are not generally invertible.

Sound Models can be used in a similar way. A target sound is compared to sounds that a model can generate, and the sound model that can "best" approximate the target sound is used to label the target sound. What makes sound models particularly useful is that not only do they have semantic labels as names (footsteps, trains, etc), but their control parameters are also labeled, and the parameter values can be used to refine the semantics. For example, a footsteps model might have a "Speed" parameter that takes on values in units of kilometers per hour. Once the model is identified and parameterized to best fit the sound, semantics can be "read off" the model as, for example, "footsteps on gravel at 2 kilometers per hour".

Sound models also have the potential to make the semantic relationship between two different sounds explicit. If two different sounds are identified with the same model, but with different parameterizations, then it would be possible, for example, to compare two different segments of footsteps sound and draw conclusions such as that for one set, the person was walking faster and on a harder surface.

6.2 Sonification

Sonification is the use of sound to communicate information in data. Sound has advantages over visual or textual representations in "hands and eyes busy" environments such as driving and surgery. It is also possible to monitor many more audio signals simultaneously than it is to monitor multiple visual signals simultaneously.

This is an interesting area for modeling not only because of the obvious dependence on real time control capabilities, but because of the relationship between sound and semantics (Hermann, T. and Ritter, 2004). To be most effective, the semantics associated with a sound should be matched to the semantics of the signal being monitored.

7. Sound Models in Content Creation

Programming sounds is hard. Just finding the right parameters to achieve a desired sound given a model architecture is a difficult process, even for experts. Take for example the Yamaha DX-7, the classic 1983 vintage FM-based synthesizer. Common folklore in the synthesis community has it that 90% of all Yamaha DX-7 synthesizers that came in for servicing had their entire set of presets intact. If there were a way to automatically program a synthesizer based on a desired sound, it would vastly increase the material accessible to sound designers and musicians.

If the synthesis architecture must be designed as well as parameterized, the problem is far more difficult. Traditional sound designers create a single static sound at a time. A sound model is designed to capture not one, but an entire range of sounds, and furthermore, traverse the range of sounds according to a specification of control and behavior.

The design process starts with a specification of the sounds and the behaviors a model should exhibit under parameter control (for example, that footsteps should have a speed control that determines rate and toe-heel characteristics).

There are several different classes of activates that interactive sound model designers engage in:

1) Building models from elementary building blocks
    a. Analyzing an existing sound for the specific features one wants to capture for the model.
    b. If a target sound is not available, then an architecture must be built and incrementally improved to meet the design goals.

        c. Defining control and parameter mapping strategies (Wanderly and Battier 2000)
2) Using existing models
    a. Finding models in a database that can generate something close to the desired sound and parameterizing them.
        i. Defining "parameter sets"
        ii. Managing the parameter dimensionality
    b. Changing an existing model to meet similar sound goals
    c. Merging existing models to create new models (that can "morph" between sounds in the original models, for example)

8. Building models from elementary units

The building blocks for sound models are modular units typically referred to as "opcodes". Opcodes include signal generators and transformers such as oscillators, filters, distortion transforms, envelope generators, random number generators, interpolators and integrators. Units as simple as mathematical operators to those as complex as other models, can serve as opcodes. CSound (Boulanger, 2000), a free and popular programming language from MIT tracing its roots back to the 1960's work of Max Mathews at Bell Labs, contains over 450 opcodes in its library (and it is still growing). The input/output modularity of opcodes and the models they are used to construct can be visualized in a signal flow type diagram such as the one used in the graphical programming language Max/MSP (Cycling'74) and shown in Figure 2.

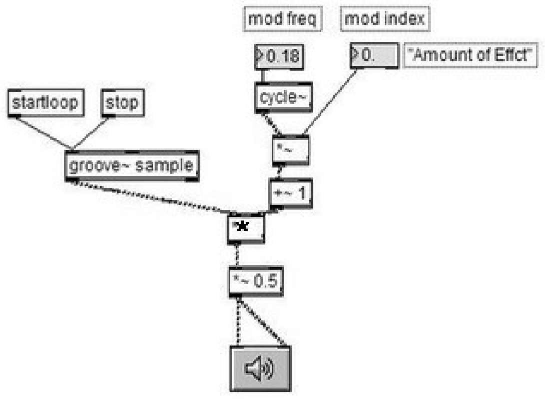

Figure 2. Opcodes units are connected to construct a sound model. This representation is directly manipulated by sound designers in the Max/MSP graphical programming language.

The process of building up sound models from opcodes is labor intensive, and requires significant programming and signal processing understanding. The graphical representation as shown in Figure 2 is a step in the right direction, but is still directly homologous to computer code. The support that graphics designers have for high-level model creation does not yet exist for audio.

Some attempts have been made to codify the knowledge that experts use to assemble models (Rolland and Pachet, 1995). Making model building reliably fast is the key to enabling most of the other applications discussed in this paper that depend on a broad collection of available models.

Often the specification of the sounds that are required to be in the model range are in the form of actual sound samples. This provides a good starting point for automated support of the model building process. Garcia (2001) for example, uses a genetic algorithm to discover model structures and parameter regimes that are capable of imitating a given target sound.

Sounds can be analyzed in a human-in-the-loop system, and specific features can be extracted and used in new sound models that may not be anything like the original sound.

An important class of sound models are physical models such as waveguides that simulate wave propagation through physical media (Smith 2002, Essl et al. 2004), and modal models that capture the resonant structure of physical objects (Gaver 1993, Doel and Pai, 1998). The models have inherent semantics and intuitive parameterizations.

8.1 Sound Model Databases for sounds

In a production environment for fixed-media such as film and television), the end goal is a specific sound, not a model. A frequently used resource is a database of recorded sound. Typically, once a sound has been found, it must still be altered in some way to fit the new context. No matter how big the database of sounds, the world of possible sounds is much larger and it would be rare to find a sound in a database fitting a designer's requirements exactly. Instead, a process of editing and digitally processing the sound typically follows.

For example, a sound designer may have a sample of a sound and simply needs to extend it in time. If the sound is textured, or contains events, simple time stretching would alter the texture. What is needed is a model of the statistical characteristics of the texture (Lu et al. 2002, Zhu and Wyse 2004). Sound textures are sounds for which there exists a window length such that the statistics of the features measured within the window stable with different window positions. That is, they are static at "long enough" time scales. Examples include crowd sounds, traffic, wind, rain, machines such as air conditioners, typing, footsteps, sawing, breathing, ocean waves, motors, and chirping birds. Since all the temporal structure exists within a determined window size, if we have a code to represent that structure for that length of time, the code is valid for any length of time greater than the texture window size.

This texture example is a kind of analysis and resynthesis, and shows that models can be more general than those that generate a class of sounds – they can also generate a class of behaviors that can be customized for a certain class of sounds given an example.

Because designers typically search a database for something close to what they are looking for and then expect to manipulate the sound to become what they want, then a database of sound models is a natural fit because they afford manipulation across a class of related sounds. The database is effectively providing the specialized tools necessary for manipulating a specific class of sounds.

A sound model can be searched in three different ways:

a) textually, using the description of the model embedded in its name and parameters,

b) sonically, using examples of sounds that each model generates

c) structurally, treating the model structure itself as the data.

Once a sound model is retrieved, it must be parameterized to produce a specific sound. Unfortunately, there are certain types of model structures that can make this task daunting.

Although it is often assumed that for a given set of parameters, a model generates a single sound, this is in practice often not true. A model that incorporates a physical model of, for example, a string instrument bowing action may have an input parameter representing the position of the bow, but internally compute the derivative of the position parameter to control sound generation. In that case, it is not the parameter value that controls the sound, but how the parameter changes in time.

Another case where a single parameter setting may map to different sounds is when the system is nonlinear. Nonlinear systems, can exhibit hysteresis where the location of a sound feature discontinuity with respect to a parameter change depends on the direction the parameter is changing.

There may be randomness built in to the model for many reasons. For example, no two footsteps sound exactly the same. If the randomness is generated internal to the model (rather than by randomness in the external control) then many different sounds can be associated with a single parameter value.

It is frequently the case that a time evolution for a sound is built in to a model, not depending upon changes to external parameters to effect the evolution. Consider the sound of a piano that dies away slowly, and even undergoes some timbral evolution although the external control is exerted only at the initiation of the note.

Even if we assume that there is only one sound associated with a given parameter set, we still have the following remaining challenges that cannot be assumed away without completely undermining the utility of the solution:

a) the parameter space can be very large,

b) many parameter settings can map to a given sound,

c) parameters change in time.

Approaches to solving the problem of searching large parameter spaces for sound synthesis include using genetic algorithms (Dahlstedt, P. 2001) or learning associations between sound features and parameters (Casey 1998).

An approach to solving the problem including time domain parameter changes is based on the its kinship to speech processing. The target sound can be broken down into a series of short time frames over which the model parameters are assumed to be fixed. Using prior learning or searching methods, a collection of candidate models and parameterizations are generated for each time frame. For each model, a cost function is defined for transitioning between parameter settings (states) for successive time frames. Finally, a Viterbi-like algorithm can be used on the sequence of candidate states to find the lowest cost path.

8.2 Sound Model Databases

Of course, if a media production is itself interactive, then a sound model must be obtained for its interactive capabilities, not just for a particular sound it makes. Because model construction is still such a laborious process, finding an existing model that at least comes close to meeting specifications could be far more efficient than building one. This is the approach taken by Funkhouser (2003) for graphics models.

Automated support for the model building process is an open problem. In graphics, a promising approach for piecing together parts of models found in a database is described in Funkhouser (2004). For graphics models, pieces are generally spatially separate regions that must then be "glued" together. For audio, it may be that different models correspond to different temporal regions of a sound that can then be glued together as is done in concatenative speech synthesis and audio mosaicing (Zils A. and Pachet, 2001). This is useful for creating a sound, but not optimal for creating an interactive model. In an interactive model, the various structural components (Figure 2), are all operative at all times. The assembly of components is a different kind of challenge than temporal assembly.

Codifying expert knowledge about assembling model parts can be helpful (Rolland and Patchet, 1995). Another approach was outlined by Altman and Wyse (2004) whereby an ontology of opcode elements is used to combine models structurally to achieve an audio "morph".

9. Summary

Interactive media elements are growing in importance and ubiquity. Sound modeling has not yet received the same kind of attention given to graphics modeling though they offer the same advantages stemming from flexibility and memory or bandwidth efficiency. The main driver is media productions that are themselves interactive such as virtual and mixed realities and computer games, but the benefits extend to other applications including fixed media production and media indexing and retrieval.

We described the components and characteristics of sound models and some of the salient open issues that need to be addressed for sound models to achieve their full potential impact.

10. Conclusions

Future work needs to focus on automated support for building sound models. If professional media developers on tight time and resource budgets are going to use models instead of traditional libraries of prerecorded sounds, then accessing (either building or finding) models must be as fast as creating or recording sounds. The time to access is the single outstanding obstacle to their widespread use, while in terms of flexibility and bandwidth, models are far superior to sounds.

For models to be useful in semantic analysis and indexing and retrieval, there must be a critical mass of models available. For large collections of models to be available, tools appropriate for the creative skills of sound designers (not signal processing engineers and programmers) will have to be available for them to become involved in constructing them. For that too, automated support for model building is the key.

11. References

[1] Altman, E. and Wyse, L. "Emergent Semantics from Media Blending", in (Srinivasan and Nepal Eds.) *Managing Multimedia Semantics.* The Idea Group Inc. (in press), 2004.

[2] Boulanger, R.C. *The CSound Book: perspectives in software synthesis, sound design, signal processing and programming.* Cambridge Mass. MIT Press, 2000.

[3] Casey, M., "Understanding Musical Sound with Forward Models and Physical Models", in Musical Networks: *Parallel Distributed Perception and Performance*, Niall Griffith and Peter M. Todd (eds.), Cambridge, MIT Press, 1998.

[4] Casey, M., "Sound Classification and Similarity Tools", in B.S. Manjunath, P. Salembier and T. Sikora, (Eds), *Introduction to MPEG-7: Multimedia Content Description Language*, J. Wiley, 2001

[5] Cycling'74 www.cycling74.com/

[6] Cook, Perry, *Real Sound Synthesis for Interactive Applications.* AK Peters, Natick. MA. 2002.

[7] Doel, K. v. d., & Pai, D. K. "The Sounds of Physical Shapes", *Presence*, *7* (4), 1998, 382–395.

[8] Essl, G., Serafin, S., Cook, P., and Smith, J.O. "Theory of Banded Waveguides", *Computer Music Journal* 28:1, 1998.

[9] Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., & Dobkin, D. "Modeling by example", *ACM Transactions on Graphics (SIGGRAPH 2004)*, to appear.

[10] Funkhouser, T. Min, P. Kazhdan, M., Chen, J., Halderman, A., Dobkin, D. and Jacobs D. "A Search Engine for 3D Models", *ACM Transactions on Graphics*, 22(1), January 2003.

[11] Garcia, R. "Automating the design of sound synthesis techniques using evolutionary methods", *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-01)*, Limerick, Ireland, December 6-8, 2001

[12] Gaver, W. W. "Synthesizing Auditory Icons", *Proceedings of the ACM INTERCHI* 1993, pp. 228–235.

[13] Hermann, T. and Ritter, H. "Sound and Meaning in Auditory Data Display", *Proceedings of the IEEE.* 92:4, April, 2004.

[14] Karplus, K., and Strong, A. "Digital Synthesis of Plucked-String and Drum Timbres", *Computer Music Journal* 7(2):43–55, 1983.

[15] Lu, L., Li, S., Liu, W., and Zhang, H. "Audio Textures*", Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2002.

[16] Rolland, P.Y., and Pachet, F. "Modeling and Applying the Knowledge of Synthesizer Patch Programmers", G. Widmer (ed.), *Proceedings of the IJCAI-95 International Workshop on Artificial Intelligence and Music, 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada, 1995.

[17] Smith, J. O. *Digital Waveguide Modeling of Musical Instruments.* Center for Computer Research in Music and Acoustics (CCRMA), Stanford University. Web published at: www-crma.Stanford.edu/~jos/waveguide/, 2002.

[18] Dahlstedt, P. "Creating and Exploring Huge Parameter Spaces: Interactive Evolution as a Tool for Sound Generation", *Proceedings of International Computer Music Conference*, Habana, Cuba, 2001.

[19] Scheirer, Eric D. "Structured Audio, Kolmogorov Complexity, and Generalized Audio Coding", *IEEE Transactions on Speech and Audio Processing.* **9**:8, Nov. 2001, pp. 914-931.

[20] Wanderly, M. and Battier, M. (eds.), *Trends in Gestural Control in Music* (Paris, France: IRCAM, 2000)

[21] Wold, E. Blum, T., Keislar, D. and Wheaton, J. "Classification, search and retrieval of audio", http://www.musclefish.com/crc/crcwin.html

[22] Wyse, L. "Free Music and the Discipline of Sound", *Organized Sound*, **8**:3, 237-247. Cambridge University Press, 2003.

[23] Wyse, L. Wang Y., Zhu X.L. Application of a Content-based Percussive Sound Synthesizer to Packet Loss Recovery in Music Streaming. *Proceeding of the 11th ACM International Conference on Multimedia* (Berkeley, CA), 2003. pp335-339.

[24] Zhu, X.L. and Wyse, L. "Sound texture modeling and time-frequency LPC", *Proceedings of the Conf. on Digital Audio Effects (DAFX-04)*, Napels, Italy, 2004. (in press).

[25] Zils A. and Pachet, F. "Musical mosaicing", *Proc. of the COST G-6 Conf. on Digital Audio Effects(DAFX-01)*, Limerick, Ireland, 2001